

A Survey of Cloud Database Systems

Ganesh Chandra Deka, *Ministry of Labor & Employment, Government of India*

This survey of 15 popular cloud databases provides an overview of each system and its storage platform, license type, and programming language used for writing the source code of the NoSQLs. It also considers features such as data-handling techniques and billing practices.

The exponential growth of the Internet has resulted in an explosion of data sources, creating storage and data-usability problems. Furthermore, an increase in the number of data types has created challenges in terms of storing and manipulating unstructured data.

These issues have lead companies and open source communities to build new tools, known as NoSQL systems or “key-value-store” systems, which aim to offer, on a massive scale, on-demand services and simplified application development and deployment. NoSQL databases are useful for applications that deal with very large semistructured and unstructured data. The growing popularity of big data will compel many companies to use NoSQL databases¹ instead of traditional database, so you can expect to see vendors offering simplified rollouts and additional support for NoSQL solutions.²

According to nosql-database.org, there are at least 150 NoSQL databases, with various features

to meet the requirements of different user groups. It’s not possible to discuss all of them here, so I selected 15 popular NoSQL databases that are representative (rather than inclusive), and I review some of their interesting characteristics.

Surveyed Systems

NoSQL databases can be divided into two groups, depending on the elasticity level. The first group contains truly elastic databases, such as MongoDB, which allows the addition of new nodes to a cluster without any observable downtime for the clients. The second group contains rigidly defined BigTable-based NoSQL databases (such as Cassandra and HBase), which have significant downtime when new nodes are added to the cluster.

Constant data availability when nodes are added or removed from the cluster is made possible by routing mechanisms and algorithms that decide when to move data chunks that are working together. For example, when data

must be moved to newly added nodes, during the copying process, the data is served from the original location. When the new node has an up-to-date version of the data, the routing processes start to send requests to this node.¹

Cassandra

The Apache Cassandra database offers good scalability and high availability without compromising performance. Its demonstrated fault-tolerance on commodity hardware (cloud infrastructures) and linear scalability make it the ideal platform for mission-critical data. Cassandra features allow replication across multiple datacenters, offering lower latency for data availability during regional outages. Cassandra's ColumnFamily information model offers the convenience of column indexes with the performance of log-structured updates, strong support for *materialize views* (also known as snapshots), and powerful built-in caching.

Netflix, Twitter, Urban Airship, Reddit, Cisco, OpenX, Digg, CloudKick, and Ooyala are some of the companies that use Cassandra to deal with huge, active, online interactive datasets. The largest known Cassandra cluster has over 300 Tbytes of information in over 400 machines (<http://cassandra.apache.org>).

BigTable

Google's BigTable maps two arbitrary string values (row key and column key) and a time stamp (creating 3D mapping) into an associated arbitrary byte array. BigTable can be characterized as a light, distributed, multidimensional sorted map. It was developed to scale to the petabyte range among numerous machines to make it easy to add machines, automatically taking advantage of those resources without any reconfiguration.³ When sizes threaten to grow beyond a specified limit, the tablets are compressed using the BMDiff^{4,5} algorithm and the Zippy open source compression algorithm,⁶ publicly known as Snappy,⁵ which is a less space-optimal variation of the LZ77 algorithm but more efficient in terms of computing time (www.aosabook.org/en/posa/infinispan.html).

To get a specific row stored in BigTable, a new client must connect to all levels of the tree, but the information obtained on the upper levels is cached, meaning that further requests for data

stored in tablets already looked for will be made directly to the last level of the tree.

HBase

HBase is an open source, distributed, versioned, column-oriented data store modeled after Google's BigTable. Basically, it's a clone of BigTable, providing a real-time, structured database on top of the Hadoop distributed file system. HBase is suitable for applications requiring random, real-time read/write access to big data. HBase's goal is to host very large tables with billions of rows and millions of columns on top of clusters of commodity hardware.

HBase provides linear and modular scalability (ability of a database to handle a growing amount of data), consistent reads and writes, automatic and configurable sharding (a horizontal partition in a database) of tables, and automatic failover support between RegionServers. It also offers convenient base classes for backing Hadoop MapReduce jobs with HBase tables, an easy-to-use Java API for client access, block cache and Bloom Filters for real-time queries, as well as query-predicate pushdown via server-side filters. Finally, it has an extensible JRuby-based shell and includes support via the Hadoop metrics subsystem to files for exporting metrics via Java Management Extensions.

MongoDB

MongoDB is an open source, schema free, document-oriented, scalable NoSQL database system. This high-performance, fault-tolerant, persistent system provides a complex query language as well as an implementation of MapReduce.

MongoDB offers

- document-oriented storage—JavaScript Object Notation (JSON)-style documents with dynamic schemas offer simplicity and power;
- full index support—that is, it can index any attribute;
- data availability—it can mirror across LANs and WANs for scalability; and
- autosharding—it scales horizontally without compromising functionality.

It also supports rich, document-based queries; atomic modifiers for contention-free performance;

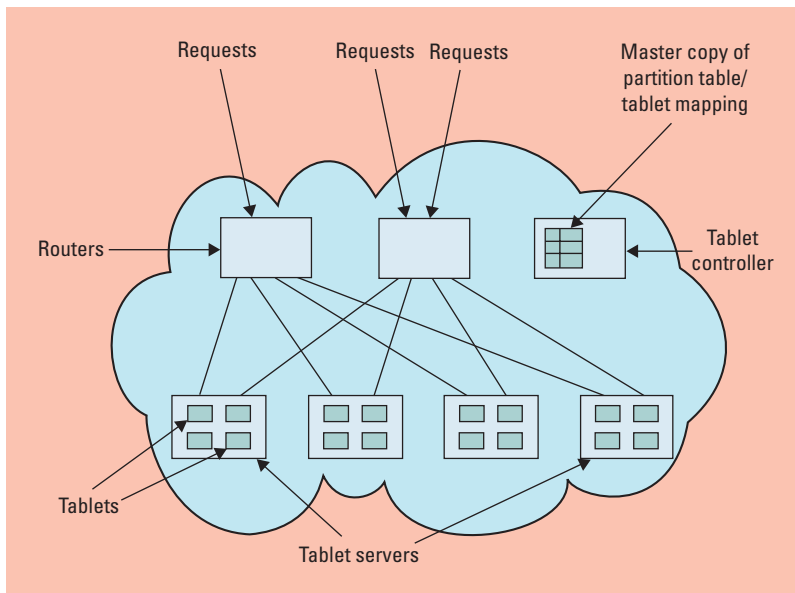


Figure 1. The Pnuts data storage architecture. Multiple applications share this massive-scale centrally managed database system.

flexible aggregation and data processing; and GridFS (a MongoDB file format for storing files larger than 16 MB), so it can store files of any size without complicating your stack.

Pnuts

The Pnuts system is a massive-scale hosted, centrally managed database system shared by multiple applications (see Figure 1). It supports Yahoo's data-serving Web applications rather than complex queries (such as offline analysis of Web crawls).⁷

Pnuts provides data management as a service. This significantly reduces application development time, because developers don't have to architect and implement their own scalable, reliable data-management solutions.⁴ Consolidating multiple applications into a single service lets users amortize operations costs over multiple applications and apply the same best practices to the data management of many different applications. Moreover, having a shared service means you can keep resources (servers, disks, and so on) in reserve and quickly assign them to applications experiencing a sudden surge in popularity.⁶

Hypertable

Hypertable is a high-performance, distributed, open source, NoSQL cloud-based data-storage system designed to support applications requiring maximum performance, scalability, and

reliability. Hypertable will be useful for organization that must administer rapidly evolving data support for online real-time applications. Modeled after Google's BigTable project, Hypertable is designed to manage information storage and processing on a large cluster of commodity servers, providing resilience to machine and component failures.

CouchDB

Apache CouchDB is a document-oriented database written using Erlang, a robust functional programming language ideal for building concurrent distributed systems. CouchDB can be queried and indexed using JavaScript in a MapReduce fashion. It also offers incremental replication with

bidirectional conflict detection and resolution. It provides a RESTful JSON API that can be accessed from any environment that allows HTTP requests. There are many third-party client libraries, making it easier to choose a programming language. CouchDB's built-in Web administration console speaks directly to the database using HTTP requests issued from the browser.

Voldemort

Voldemort is a fully distributed key-value storage system. Each node is independent with no central point of failure or coordination. Voldemort is designed for use as a simple storage that's fast enough to avoid needing a caching layer on top of it. The software architecture is made of several layers, each of which implements the *put*, *get*, and *delete* operations. Each layer is responsible for a specific function, such as TCP/IP communications, routing, or conflict resolution.

With Voldemort, data is automatically replicated over multiple servers as well as automatically partitioned, so each server contains only a subset of the total data. Server failure is handled transparently, and pluggable serialization is supported to allow rich keys and values, including lists and tuples with named fields. Voldemort can be integrated with common serialization frameworks, such as Protocol Buffers, Thrift, Avro, and Java Serialization.

Furthermore, in Voldemort,

- data items are versioned to maximize data integrity in failure scenarios without compromising system availability;
- single node performance is capable of 10,000–20,000 operations per second, depending on the machines, network, disk system, and data-replication factor; and
- pluggable data placement strategies can support features such as distribution across datacenters that are geographically far apart.

Voldemont is used in LinkedIn for high-scalability storage features, where simple functional partitioning isn't sufficient. However, it often has error messages, likely caused by uncaught bugs. The source code is available under the Apache 2.0 license.⁵

Infinispan

Infinispan is a peer-to-peer, in-memory data grid (IMDG) platform, written for the Java virtual machine (www.aosabook.org/en/posa/infinispan.html). It's an extremely scalable, highly available, open source data grid platform. It's a distributed, in-memory key-value NoSQL store, designed to make the most of up-to-date multiprocessor/multicore architectures and to provide distributed cache capabilities. Infinispan presents a cache interface, extending `java.util.Map` as well as optionally backing a peer-to-peer network architecture to distribute the system efficiently around a data grid.

Dynomite

Dynomite provides integrated storage and distribution, requiring developers to adopt a simple, key-value data model to improve availability and scalability. By separating availability and scalability, developers can take advantage of Dynomite's sophisticated distribution and scaling techniques while still having great flexibility in choosing the data model (<https://github.com/cliffmoon/dynomite/wiki/dynomite-framework>).

Written in Erlang, Dynomite is a consistent, distributed, key-value-store NoSQL database system. The design is based on Amazon's Dynamo paper.⁸ Dynomite currently implements vector clocks, Merkle trees, and consistent hashing. It also supports tunable quorum sensing,

membership gossiping, gossiped synchronization of partitions, pluggable storage engines, a thrift interface, and a Web console with canvas visualizations.

Redis

Redis (*remote dictionary server*) is an easy-to-use open source, advanced key-value in-memory database in which all keys stay in memory (RAM). It's often referred to as a data-structure server, because keys can contain strings, hashes, lists, sets, and sorted sets.⁹ Its node to node protocol is binary, optimized for bandwidth and speed, but its values can't be bigger than 512 Mbytes, and it doesn't use the operating system's virtual memory.

Xeround

Xeround offers its own elastic database service based on MySQL. It hosts the service at Amazon Web Services datacenters located in Europe and North America, so customers can choose whichever location is closest.¹⁰ Xeround's database is host agnostic, so users can easily migrate between cloud service providers.

Xeround's two-tier architecture comprises access nodes and data nodes. Access nodes receive application requests, communicate with data nodes, perform computations, and deliver request results, while data nodes store the data. Xeround stores data in virtual partitions that aren't bound to the underlying hardware infrastructure. Each partition is replicated to the different data nodes, located on separate servers, providing high availability and full resiliency. In addition to offering multiple geographic locations, Xeround is planning to offer its services to other cloud providers, including GoGrid and Rackspace.

Xeround Basic offers data sizes up to 500 Mbytes, supports up to 40 concurrent connections (Xeround Pro supports up to 4,800 connections), and has a max throughput up to 8 Mbytes per second.

SimpleDB

Amazon Web Services has its own NoSQL cloud database service called SimpleDB, which can support basic use cases, and minimal use is free. The SimpleDB source code is written using Erlang, allowing for flexible design, easy scalability, and extensibility.⁴

Dynamo

Amazon Dynamo helps manage the state of services that have high reliability requirements and require tight control over the tradeoffs between availability, consistency, cost-effectiveness, and performance. Dynamo provides a simple primary-key only interface to accommodate application requirements. It uses a combination of successfully implemented techniques to accomplish scalability and availability: data is abstracted and consistently replicated¹¹ through object versioning.¹² Replica consistency is maintained during updates using a synchronization protocol that applies a quorum-like technique and decentralized replica. Dynamo provides a decentralized arrangement, keeping administration charges to a minimum. Storage nodes are automatically added and removed without requiring manual administration or redistribution.

Dynamo has been the fundamental storage technology for several core services in Amazon's e-commerce platform. It can scale to extreme peak loads efficiently without any downtime during holiday shopping seasons. For example, the service that maintains the shopping cart serves millions requests, which have resulted in over three million checkouts in a single day, and the service that manages the session state has handled thousands of concurrently active sessions.⁹

ClearDB

ClearDB also offers a hosted relational database. This globally distributed MySQL database has high availability, survivability, and performance.¹³ It offers multiregional read/write mirroring with 100 percent uptime, even if networks or disks fail. It's also cloud agnostic—that is, its hybrid configuration can span over multiple clouds as well as physical datacenters scattered over a wide geographical area.

System Capability Comparisons

Here, I consider the systems based on two main features.

Data-Handling Techniques

BigTable and similar sort systems, such as Cassandra and HBase, attempt to perform sequential I/O for updates, because this approach never overwrites records on a disk. Instead,

updates are written to a buffer in memory, and the entire buffer is written sequentially to disk. Multiple updates to the same record can be flushed at different times to different parts of the disk. As a result, to read a record, BigTable type NoSQL databases must perform multiple I/Os to retrieve and combine the various updates. Because all write operations are sequential, the writes are fast, but reads are correspondingly deoptimized.

In contrast, the traditional buffer-pool architecture of Pnuts overwrites records when they're updated. Because updates require random I/O, they're comparatively slower than BigTable and similar systems, but reads are fast because a single I/O can retrieve the entire latest record.

HBase doesn't synchronize log updates to disk, which results in low latency updates and high throughput. This is appropriate for HBase's target use—to run batch analytics on serving data, rather than to present guaranteed robustness for such data. For such a system, high-throughput sequential reads and writes are favored over durability for random updates. Pnuts always forces log updates to a disk when committing a transaction, although this log force can be disabled.

Cassandra and Pnuts support asynchronous replication—that is, wide-area replication—without adding significant overhead to the update call itself. In this model, writes are allowed anywhere, and conflicting writes to the same object are resolved afterward.

Synchronous reproduction ensures freshness of replicas and is used in HBase and Cassandra.

Column storage is beneficial for an application that must access a known subset of columns for each request. BigTable, HBase, and Cassandra all provide the capability to declare column groups or families and to add columns. Each group/family is physically stored separately. On the other hand, if requests typically want the entire row, or arbitrary subsets of it, partitioning that keeps the entire row physically together is best. This can be done with row storage in Pnuts or by using a single column group/family in a column store like Cassandra.

To conclude, *get*, *put*, and *delete* functions are best supported by key-value systems, such as Hypertable and Voldemort, while data *aggregation*

becomes much easier using column-oriented NoSQLs (such as Cassandra or BigTable). *Mapping* data becomes easier using document-oriented NoSQL databases, such as MongoDB.

Billing Practices

Cassandra lets client specify on a per-call basis whether the write is durably persistent. Amazon DynamoDB is a service from Amazon that provides NoSQL database service with seamless scalability. It lets a user launch a new Amazon DynamoDB database table and scale the request capacity up or down for the table without downtime or performance degradation. With Amazon Elastic Compute Cloud (EC2), a user can get root access but pays for idle time, when no computation is being done. Users can calculate the cost of computing in the Amazon website at <http://calculator.s3.amazonaws.com/calc5.html>.

Each month, Amazon SimpleDB users pay no charges on the first 25 machine hours and 1 Gbyte of storage consumed. Amazon DynamoDB users pay no charges on the first 100 Mbytes of storage, as well as five writes per second and 10 reads per second of ongoing throughput capacity (<http://calculator.s3.amazonaws.com/index.html>).

MongoDB subscription rates are as follows: the basic subscription is US \$2,500 per server, the standard subscription is \$5,000 per server, and the enterprise subscription is \$7,500 per server (www.mongodb.com/products/mongodb-subscriptions/pricing).

Memory (RAM) is one of the key resources that NoSQL databases heavily rely on for performance boosts. By sharing RAM across multiple servers, NoSQL picks up the ability for easy scale-out operations while also benefiting from increased redundancy or higher availability through fault tolerance. NoSQL's scale-out strategy also picks up additional cost benefits, because most implementations are open source. In fact, almost all the NoSQLs discussed here are either open source or have an open source version with limited features.

Organizations planning to use NoSQL databases must understand their storage requirements as well as the long-term implications, in addition to the cost factor. Some of the factors for using NoSQLs include

- large, unstructured, sparse, and growing data;
- a less rigid schema; and
- performance and availability are preferred over redundancy.

Another important factor is cost effectiveness—that is, when clusters of cheap commodity servers are used to manage the exploding volume of data and transaction.

Observations

Advancements in the NoSQL architecture motivated Yahoo to develop criteria for quantitatively evaluating NoSQL systems. Its Cloud Servicing Benchmark is a well-known benchmarking framework used to evaluate many different NoSQL databases, and it can be extended to support varying workload levels (<http://code.google.com/appengine/docs/python/runtime.html>).

Table 1 lists the salient features of the 15 systems discussed in this article.

The two main issues for selecting a particular database are

- addressing the workload requirements, which involves comparing read-optimized versus write-optimized substitution; and
- weighing latency versus durability (for example, if developers know that they can lose a small fraction of writes—such as Web poll votes—then they can acknowledge successful writes without waiting for them to be synched to the disk).

Database.com is a database engine for cloud application developers. It's a pilot project developed by Salesforce.com to power its website. It offers high scalability and availability; a relational data store, suitable for many enterprise needs; file storage for documents, videos, and images; SOAP and REST APIs; and various toolkits for Java, .NET, Ruby, Python, iOS, Android, Google App Engine, Google Data, Microsoft Azure, Amazon Web Services, Facebook, Twitter, and Adobe Flash/Flex.¹⁴

RedHat has transformed the world of data-storage software the way it revolutionized the market for the Unix-based operating system (www.redhat.com/database_availability). Use of


Table 1. Salient features of the 15 NoSQL systems surveyed in this article.*

NoSQL system	Storage type/platform	License type	Programming language
Cassandra	Column	Open source	Java
BigTable	Column	Proprietary	C
HBase	Column	Open source	Java
MongoDB	Document	Open source/ General Public License	C++
Pnuts	Column	Proprietary	Java/Java virtual machine
Hypertable	Key-value	General Public License/ open source	C++
CouchDB	Document	Open source	Erlang
Voldemort	Key-value	Open source	Java
Infinispan	Data grid cloud	Open source	Java
Dynomite	Key-value	Open source	Erlang
Redis	Key-value/tuple	Open source	C
Xeround	MySQL based	General Public License/ open source	C/C++†
SimpleDB	Document	Proprietary	Erlang
Dynamo	Key-value	Open source	Erlang
ClearDB	MySQL based	Open source	C†

*Source: <http://nosql.findthebest.com> and <http://nosql-database.org>

†Developed in C or C++

multiple cloud infrastructures, coupled with the automation of PaaS, will make cloud databases services the natural choice, preferred over do-it-yourself or managed databases.¹¹

Data management has become critical, considering the wide range of storage locations and plethora of mobile devices. Data has already escaped from the IT department's control, moving into the wider reaches of cloud-based services, mobile devices, and social networking sites. The proliferation and maturity of cloud computing will increase the need for reliable cloud-database services. Of the 15 cloud databases surveyed here, Cassandra, HBase, and MongoDB are the most widely used and thus the most representative of the NoSQL world. Instead of supporting ACID (atomicity, consistency, isolation, durability) transactions, many cloud databases support BASE (basically available, soft state, eventual consistency) principles. As an extension of this work, I plan to perform a "BASE" analysis of Cassandra, HBase, and MongoDB. 

References

1. T. Dory, "Study and Comparison of Elastic Cloud Databases: Myth or Reality?" master's thesis, Computer Eng. Dept., Université Catholique de Louvain, 2011; www.info.ucl.ac.be/~pvr/MemoireThibaultDory.pdf.
2. S. Sakr et al., "A Survey of Large Scale Data Management Approaches in Cloud Environments," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 3, 2011, pp. 311–336.
3. "Xeround—Cloud Database Frequently Asked Questions (FAQs)," GetApp.com, 2014; www.getapp.com/q/xeround-cloud-database-application.
4. "Major Technology Players Look to Open Source Model for New Data Center Project," EnterpriseDB.com, 27 Dec. 2011; <http://enterprisedb.com/news-events/news/major-technology-players-look-open-source-model-new-data-center-project>.
5. J. Rogers, O. Papaemmanouil, and U. Cetintemel, "A Generic Auto-Provisioning Framework for Cloud Databases," *Proc. IEEE 26th Int'l Conference Data Eng. Workshops (ICDEW 10)*, IEEE, 2010, pp. 63–68; doi: 10.1109/ICDEW.2010.5452746.
6. F. Chang et al., "Bigtable: A Distributed Storage System for Structured Data," *ACM Trans. Computer Systems*, vol. 26, no. 2, 2008, article 4; doi: 10.1145/1365815.1365816.
7. B.F. Cooper et al., "Benchmarking Cloud Serving Systems with YCSB," *Proc. 1st ACM Symp. Cloud Computing (SoCC 10)*, 2010, pp. 143–154; http://ipij.aci.polsl.pl/django-media/lecture_file/ycsb.pdf.

8. G. DeCandia et al., "Dynamo: Amazon's Highly Available Key-Value Store," *Proc. 21st ACM SIGOPS Symp. Operating Systems Principles (SOSP 07)*, 2007, pp. 205–220; doi: 10.1145/1294261.1294281.
9. A. Masudianpour, "An Introduction to Redis Server, An Advanced Key Value Database," SlideShare, 9 Aug. 2013; www.slideshare.net/masudianpour/redis-25088079.
10. N. Vekiarides, "Ten Hot Trends in Cloud Data for 2012," *Cloud Computing J.*, 3 Jan. 2012; <http://cloudcomputing.sys-con.com/node/2109293>.
11. D.C. Giuseppe et al., "Dynamo: Amazon's Highly Available Key-Value Store," *Proc. 21st ACM Symp. Operating Systems Principles (SOSP-07)*, 2007, pp. 205–220.
12. L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System," *Comm. ACM*, vol. 21, no. 7, 1978, pp. 558–565.
13. C. Coleman, "Welcome to the New ClearDB," *ClearDB*, 17 Oct. 2011; www.cleardb.com/blog/entry?id=beta/welcome.
14. K. Subramanian, "Database.com: Salesforce's New RDBMS as a Service Offering," *Cloud Ave.*, 6 Dec. 2010; www.cloudave.com/8542/database-com-salesforces-new-rdbms-as-a-service-offering.

Ganesh Chandra Deka is an assistant director of training at DGE&T, Ministry of Labor & Employment, Government of India. His research interests include cloud computing, e-governance, and speech processing. Contact him at ganeshdeka2000@gmail.com.



Selected CS articles and columns are available for free at <http://ComputingNow.computer.org>.

Call for Articles

IEEE Software seeks practical, readable articles that will appeal to experts and nonexperts alike. The magazine aims to deliver reliable information to software developers and managers to help them stay on top of rapid technology change. Submissions must be original and no more than 4,700 words, including 200 words for each table and figure.

**IEEE
Software**

Author guidelines: www.computer.org/software/author.htm
Further details: software@computer.org
www.computer.org/software